AMENDMENTS TO THE CLAIMS

1. (Currently amended) In a computer system having an operating environment including user mode modules having a first level of protection and kernel mode modules having a second level of protection, a method for consistently collecting information associated with the execution of a user mode module, the method comprising:

transmitting, by a requestor application, a request to collect kernel mode module information, wherein the request to collect kernel mode module information includes an identification of one or more executing process threads from which kernel mode information will be collected;

obtaining, by a kernel mode module, corresponding to a driver application external to the operating system, the request to collect kernel mode module information;

capturing, by the kernel mode module, information corresponding to each thread identified in the request to collect kernel mode module information;

transmitting, by the kernel mode module, a result of the capturing of the information corresponding to each thread identified in the request to collect kernel mode module information; and

receiving, by the requestor application, the result of the capturing of the information corresponding to each thread identified in the request to collect kernel mode module information.

2. (Original) The method as recited in Claim 1, wherein the request to capture kernel mode module information includes an identification of a pre-allocated memory in which to store captured kernel mode information.

3. (Canceled)

4. (Previously presented) The method as recited in Claim 1, wherein the kernel mode module is an operating system resident application.

5. (Original) The method as recited in Claim 1 further comprising capturing, by the kernel mode module, a list of all loaded drivers.

6. (Original) The method as recited in Claim 1, wherein capturing information corresponding to each thread identified in the request to collect kernel mode module information includes:

(a) capturing a thread kernel stack; and

(b) capturing all pending I/O request packet information; and

(c) repeating (a) – (b) for each identified thread in the request to capture kernel mode module information.

7. (Original) The method as recited in Claim 6, wherein capturing all pending I/O request packet information includes:

(a) capturing an identification of all pending I/O request packets;

(b) capturing current stack location for the identified I/O requests;

(c) capturing device object information;

(d) capturing file object information;

(e) capturing driver object information; and

(f) repeating (a) – (e) for each I/O request packet corresponding to a current thread.

8. (Original) The method as recited in Claim 6, wherein capturing information corresponding to each thread identified in the request to collect kernel mode module information is asynchronous.

9. (Original) The method as recited in Claim 1, wherein transmitting a result includes transmitting a status code corresponding to the success or failure of the information capture.

10. (Original) The method as recited in Claim 1, wherein transmitting a result includes storing the captured kernel mode module information in an allocated memory.

11. (Original) The method as recited in Claim 1, wherein transmitting a request to collect kernel mode module information occurs in response to a user mode module error.

12. (Original) A computer-readable medium having computer-executable instructions for performing the method recited in Claim 1.

13. (Original) A computer system having a processor, a memory and an operating environment, the computer system for performing the method recited in Claim 1.

14. (Currently amended) In a computer system having an operating environment including user mode modules having a first level of protection and kernel mode modules having a second level of protection, a method for consistently collecting information associated with the execution of a user mode module, the method comprising:

obtaining a user mode module request to collect kernel mode module information including an identification of one or more executing process threads from which kernel mode information will be collected;

wherein obtaining a user mode module request includes obtaining, by a driver application external to the operating system, the user mode module request;

capturing information corresponding to each thread identified in the request to collect kernel mode module information; and

transmitting the captured kernel mode module information.

15. (Original) The method as recited in Claim 14, wherein the request to capture kernel mode module information includes an identification of a pre-allocated memory in which to store captured kernel mode information.

16. (Canceled)

17. (Original) The method as recited in Claim 14, wherein obtaining a user mode module request includes obtaining, by an operating system resident application, the user mode module request.

18. (Original) The method as recited in Claim 14 further comprising capturing a list of all loaded drivers.

19. (Original) The method as recited in Claim 14, wherein capturing information corresponding to each thread identified in the request to collect kernel mode module information includes:

(a) capturing a thread kernel stack; and

(b) capturing all pending I/O request packet information; and

(c) repeating (a) – (b) for each identified thread in the request to capture kernel mode module information.

20. (Original) The method as recited in Claim 19, wherein capturing all pending I/O request packet information includes:

(a) capturing an identification of all pending I/O request packets;

(b) capturing current stack location for the identified I/O request;

(c) capturing device object information;

(d) capturing file object information;

(e) capturing driver object information; and

(f) repeating (a) – (e) for each I/O request packet corresponding to a current thread.

21. (Original) The method as recited in Claim 19, wherein capturing information corresponding to each thread identified in the request to collect kernel mode module information is asynchronous.

22. (Original) The method as recited in Claim 1, wherein transmitting the captured kernel mode module information includes transmitting a status code corresponding to the success or failure of the information capture.

23. (Original) A computer-readable medium having computer-executable instructions for performing the method recited in Claim 14.

24. (Original) A computer system having a processor, a memory and an operating environment, the computer system for performing the method recited in Claim 14.

25. (Currently amended) In a computer system having a processor, a memory, and an operating environment, the operating environment including user mode modules having a first level of protection and kernel mode applications having a second level of protection, ~~a software architecture for consistently collecting information associated with the execution of a user mode module, the architecture~~ the system comprising:

a driver application comprising a processing component for capturing kernel mode module information corresponding to one or more executing processing threads identified in a request to collect kernel mode module information; and

at least one application program interface for accessing the processing component and identifying the one or more executing processing threads from which to collect kernel mode module information.

26. (Original) The architecture as recited in Claim 25, wherein the at least one application program interface is further operable to identify a pre-allocated memory to received captured kernel mode module information.

27. (Canceled)

28. (Original) The architecture as recited in Claim 25, wherein the processing component is embodied as an operating system resident application.

29. (Original) The architecture as recited in Claim 25, wherein the kernel mode module information includes a list of all loaded drivers.

30. (Original) The architecture as recited in Claim 25, wherein the kernel mode module information includes a thread kernel stack and all pending I/O request packet information for each identified process thread.

31. (Previously presented) The architecture as recited in Claim 30, wherein all pending I/O request packet information includes an identification of the pending I/O request packet, a current stack location, device object information, file object information and driver object information for each identified I/O request packet corresponding to an identified process thread.

32. (Original) The architecture as recited in Claim 25, wherein the process component captures the kernel mode module information asynchronously.